C²I² Systems

CCII Systems (Pty) Ltd Registration No. 1990/005058/07

Communications
Computer Intelligence
Integration

# User Manual

**for the**

# FDDI Adapter

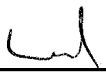
# VxWorks Enhanced Network Software Driver

| | |
|---|---|
| **C²I² Systems Document No.** | CCII/FDDI/6-MAN/001 |
| **Document Issue** | 1.3 |
| **Issue Date** | 2009-08-20 |
| **Print Date** | 2009-08-20 |
| **File Name** | P:\FDDI\TECH\MAN\CFDMAN01.WPD |
| **Distribution List No.** | |

*© C²I² Systems The copyright of this document is the property of C²I² Systems. The document is issued for the sole purpose for which it is supplied, on the express terms that it may not be copied in whole or part, used by or disclosed to others except as authorised in writing by C²I² Systems.*

*Document prepared by C²I² Systems, Cape Town*

# Signature Sheet

| Name | Signature | Date |
|---|---|---|
| Completed by<br>pp. X. Kruger | <br>Project Engineer<br>Board Level Products<br>C²I² Systems | 2009-08-20 |
| Accepted by<br>L W DE LANGE |  <br>Project Manager<br>Board Level Products<br>C²I² Systems | 2009-08-28 |
| Accepted by<br>X. Kruger | <br>Quality Assurance<br>C²I² Systems | 2009-08-20 |

# Signature Sheet

| Name | Signature | Date |
|---|---|---|
| Completed by | | |
| | Project Engineer<br>Board Level Products<br>C²I² Systems | |
| Accepted by | | |
| | Project Manager<br>Board Level Products<br>C²I² Systems | |
| Accepted by | | |
| | Quality Assurance<br>C²I² Systems | |

# Amendment History

| Issue | Description | Date | ECP No. |
|---|---|---|---|
| 1.0 | Initial version created for END Driver review. | 2000-06-21 | - |
| 1.1 | Updated to include support for multiple cards per single host. | 2002-04-03 | - |
| 1.2 | Update to include description of command line argument passing. | 2002-07-26 | - |
| 1.3 | Improve document naming consistency. | 2009-08-20 | CCII/FDDI/6-ECP/034 |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# Contents


1. **Scope** . . . 1
   - 1.1 Identification . . . 1
   - 1.2 Introduction . . . 1

2. **Applicable and Reference Documents** . . . 2
   - 2.1 Applicable Documents . . . 2
   - 2.2 Reference Documents . . . 2

3. **Installation Procedure** . . . 3
   - 3.1 To Build the FDDI (END) VxWorks Software Driver Into the VxWorks Kernel . . . 3
   - 3.2 To Load the Driver Software Separately . . . 4
   - 3.3 Starting the Driver . . . 4
     - 3.3.1 Command Line Passing of FDDI Settings . . . 4
   - 3.4 Attaching Driver to TCP/IP Stack . . . 5

4. **Application Program Interface (API)** . . . 6
   - 4.1 Driver Functions . . . 6
     - 4.1.1 void ccfddiGetStats(int unit, struct cc_fddi_mib_type *data) . . . 6
     - 4.1.2 void ccfddiClrStats(int unit) . . . 6
   - 4.2 Driver Data Structures . . . 6
     - 4.2.1 cc_fddi_mib_type . . . 6
   - 4.3 Application Example . . . 7

5. **Getting Started** . . . 8

6. **Contact Details** . . . 10
   - 6.1 Contact Person . . . 10
   - 6.2 Physical Address . . . 10
   - 6.3 Postal Address . . . 10
   - 6.4 Voice and Electronic Contacts . . . 10
   - 6.5 Product Support . . . 10

# Abbreviations and Acronyms

| | |
|---|---|
| BIT | Built-in Test |
| BSD | Berkley Software Distribution |
| BSP | Board Support Package |
| END | Enhanced Network Device |
| FDDI | Fibre Distributed Data Interface |
| ICMP | Internet Control Message Protocol |
| IP | Internet Protocol |
| LAN | Local Area Network |
| MIB | Management Information Base |
| NIC | Local Area Network |
| PC | Personal Computer |
| PCI | Peripheral Component Interconnect |
| PMC | Peripheral Component Interconnect Mezzanine Card |
| SENS | Scalable Enhanced Network Stack |
| SMT | Station Management |
| TCP | Transmission Control Protocol |
| VME | Versa Module Eurocard |

# 1. Scope

## 1.1 Identification

This document is the User Manual for the C²I² Systems Fibre Distributed Data Interface (FDDI) Enhanced Network Device (END) VxWorks Software Driver.

## 1.2 Introduction

The FDDI (END) VxWorks Software Driver was developed specifically to operate on DY4's SVME178 and Radstone's PPC1a host carrier cards and Intel X86 PC platforms. As such the drivers binaries are provided with explicit installation instructions. The FDDI (END) VxWorks Software Driver conforms to the VxWorks 5.4 (END) driver model and the driver will interface to VxWorks using the standard.

The FDDI adapters attach computers to 100 Mbit/s FDDI networks using fibre optic cable.

The driver software distribution consists of the following files :

| | |
|---|---|
| ccFddiEndxxx.a[1] | FDDI driver object file |
| Readme.txt | Installation notes |
| Release.txt | Release notes and revision history : Please check this file for information on the latest updates |
| ccMib.h | Defines the structure cc_fddi_mib_type for accessing the FDDI Management Information Base (MIB) |
| ShowStat.c | Sample C program for accessing the FDDI MIB. |

[1] *xxx*=178 for SVME178 PPC, X86 for Intel X86 or Rad for Radstone PPC1a

## 2. Applicable and Reference Documents

### 2.1 Applicable Documents

2.1.1 DI-IPSC-81443, *Data Item Description for a Software User Manual*.

2.1.2 ISO/IEC 9314-6 / X3.229, *Information technology -- Fibre Distributed Data Interface (FDDI) -- Part 6: Station Management (SMT)*, 7 October 1993.

2.1.3 CCII/A500/IMS/6-MAN/1, *PMC FDDI Network Card Installation Guide*.

2.1.4 *VxWorks 5.4 Programmer's Guide*, edition 1.

2.1.5 RFC-1512, *FDDI Management Information Base*, dated September 1993.

### 2.2 Reference Documents

2.2.1 ANSI INCITS 229, *Information Systems - Fibre Distributed Data interface (FDDI) - Station Management (SMT)*, dated 1994-01-01.

## 3. Installation Procedure

This paragraph describes the installation instructions for the FDDI (END) VxWorks Software Driver.

### 3.1 To Build the FDDI (END) VxWorks Software Driver Into the VxWorks Kernel

1. Copy ccFddiEnd*xxx*.a[1] to your Board Support Package (BSP) library directory (eg /tornado/target/config/svme178/lib).

2. Edit the Makefile in the BSP directory ( eg /tornado/target/config/svme178).

   Find the line
   ```
   MACH_EXTRA  =
   ```
   and replace with
   ```
   MACH_EXTRA  =      ./lib/ccFddiEndxxx.a
   ```

3. Add the following code fragment to config.h.

   (before "`#define DEFAULT_BOOT_LINE`") :

   ```
   /*Added by CCII for FDDI driver*/
   #define INCLUDE_CCFDDI_END     /* CCII FDDI END DRIVER */

   /* CCII Modification */
   #ifdef INCLUDE_CCFDDI_END
   #ifndef INCLUDE_PCI
   #define INCLUDE_PCI
   #endif

   #ifdef INCLUDE_BSD
   #undef INCLUDE_BSD
   #endif

   #ifndef INCLUDE_END
   #define INCLUDE_END
   #endif
   #endif

   /*End  CCII Modification */
   ```

4. In config.h, change the `DEFAULT_BOOT_LINE` to use the ccfddi driver.

5. Add the folowing to configNet.h.

   (Before the start of the endDevTbl[] decleration.)

   ```
   IMPORT END_OBJ * fddiLoad (char *initString);
   ```

   (Add the following segment to the endDevTbl[] , before the default last entry.)

   ```
   #ifdef INCLUDE_CCFDDI_END
      { 0, fddiLoad, "", NULL, NULL, FALSE},
   #endif /* INCLUDE_CCFDDI_END */
   ```

6. Rebuild all VxWorks images.

[1] *xxx*=178 for SVME178 PPC, 179 for SVME179, X86 for Intel X86 or Rad for Radstone PPC1a

## 3.2 To Load the Driver Software Separately

From the VxWorks shell, type :

`ld < ccFddiEnd`*xxx*`.a`[1]

## 3.3 Starting the Driver

The driver is started and attached to the MUX with the VxWorks `muxDevLoad`and `muxDevLoad` commands. The syntax is as follows:

`muxDevStart ( muxDevLoad ( `*unitno*`, fddiLoad,"", 0, 0 ) )`

where *unitno* should usually be 0(Zero)[2]. If two FDDI cards are installed on the same host, unitno=0 for the first card and unitno=1 for the second card. This is only necessary if the driver was not included in the configNet.h file as described in paragraph 3.1.

Type `muxShow` to see if the driver was installed.

### 3.3.1 Command Line Passing of FDDI Settings

The FDDI Driver supports the changing of some default parameters at startup.
e.g. muxDevStart (muxDevLoad (0,fddiLoad,"BaseAddress:IntoNo:TRT:MTU:OSM",0,0))

| | | |
|---|---|---|
| BaseAddress | : | Base Address in Hex - Only has effect on MVME2700 |
| IntNo | : | Interrupt no to use in Hex - Only has effect on MVME2700 |
| TRT | : | Token Rotation Time to use in Decimal ms - Default 165ms |
| MTU | : | Max Transfer Unit size in Decimal bytes - Default 4352bytes |
| OSM | : | Online Special Mode ON(1) or OFF (0) - Default 0 (OFF) |

If a parameter is not used a placeholder of 0 MUST be used.
e.g. You want to only change the TRT to 100ms
muxDevStart (muxDevLoad (0,fddiLoad,"0:0:100:0:0",0,0))

All Parameters preceding the ones you want to change must be specified.
e.g. You want to only change the TRT to 100ms
muxDevStart (muxDevLoad (0,fddiLoad,"0:0:100",0,0))

To keep all setting at default values use :

muxDevStart ( muxDevLoad ( unit, fddiLoad, "", 0, 0 ))

Use ccfddiStats(int unit) to confirm your changes.

---

[1] *xxx*=178 for SVME178 PPC, 179 for SVME179, X86 for Intel X86 or Rad for Radstone PPC1a

[2] Known Bug In SENS 1 under Tornado 1 :
You can not have more than one driver with the same *unitno*. So if you have a ethernet driver installed as ether0, you can not have a ccfddi0, but must make the *unitno* 1. This will change it to ccfddi1. This bug has been fixed in Tornado 2.

## 3.4 Attaching Driver to TCP/IP Stack

Attach the Driver to the TCP/IP protocol stack with :

```
ipAttach unitno, "ccfddi"
```

Use `muxShow` and `ifShow` to confirm.

# 4. Application Program Interface (API)

## 4.1 Driver Functions

### 4.1.1 void ccfddiGetStats(int unit, struct cc_fddi_mib_type *data)

This function is allows an application to access information about the driver as stored in the MIB. If two FDDI cards are installed on the same host, unitno=0 for the first card and unitno=1 for the second card.

### 4.1.2 void ccfddiClrStats(int unit)

This function resets the following counters in the FDDI MIB :

- fddiMACFrame-Ct
- fddiMACCopied-Ct
- fddiMACTransmit-Ct
- fddiMACError-Ct
- fddiMACLost-Ct
- fddiPORTLCTFail-Ct
- fddiPORTLem-Reject-Ct
- fddiPORTLem-Ct

If two FDDI cards are installed on the same host, unitno=0 for the first card and unitno=1 for the second card.

## 4.2 Driver Data Structures

### 4.2.1 cc_fddi_mib_type

```
struct cc_fddi_mib_type
{
unsigned int        rx_packets;           /* total packets received */
unsigned int        tx_packets;           /* total packets transmitted */
unsigned int        rx_errors;            /* bad packets received */
unsigned int        tx_errors;            /* packet transmit problems */
unsigned int        rx_dropped;           /* no space in linux buffers */
unsigned int        tx_dropped;           /* no space available in VxWorks */
unsigned int        multicast;            /* multicast packets received */
unsigned int        transmit_collision;   /* always 0 for FDDI */

/* Detailed FDDI statistics. */

unsigned char       smt_station_id[8];
            unsigned int           smt_op_version_id;
            unsigned int           smt_hi_version_id;
            unsigned int           smt_lo_version_id;
            unsigned char          smt_user_data[32];
            unsigned int           smt_mib_version_id;
            unsigned int           smt_mac_cts;
            unsigned int           smt_non_master_cts;
            unsigned int           smt_master_cts;
            unsigned int           smt_available_paths;
            unsigned int           smt_config_capabilities;
            unsigned int           smt_config_policy;
            unsigned int           smt_connection_policy;
            unsigned int           smt_t_notify;
            unsigned int           smt_stat_rpt_policy;
            unsigned int           smt_trace_max_expiration;
            unsigned int           smt_bypass_present;
            unsigned int           smt_ecm_state;
            unsigned int           smt_cf_state;
            unsigned int           smt_remote_disconnect_flag;
            unsigned int           smt_station_status;
            unsigned int           smt_peer_wrap_flag;
            unsigned long int      smt_time_stamp;
            unsigned long int      smt_transition_time_stamp;
            unsigned int           mac_frame_status_functions;
            unsigned int           mac_t_max_capability;
            unsigned int           mac_tvx_capability;
            unsigned int           mac_available_paths;
            unsigned int           mac_current_path;
            unsigned char          mac_upstream_nbr[CC_ALEN];
            unsigned char          mac_downstream_nbr[CC_ALEN];
            unsigned char          mac_old_upstream_nbr[CC_ALEN];
```

```
unsigned char          mac_old_downstream_nbr[CC_ALEN];
unsigned int           mac_dup_address_test;
unsigned int           mac_requested_paths;
unsigned int           mac_downstream_port_type;
unsigned char          mac_smt_address[CC_ALEN];
unsigned int           mac_t_req;
unsigned int           mac_t_neg;
unsigned int           mac_t_max;
unsigned int           mac_tvx_value;
unsigned int           mac_frame_cts;
unsigned int           mac_copied_cts;
unsigned int           mac_transmit_cts;
unsigned int           mac_error_cts;
unsigned int           mac_lost_cts;
unsigned int           mac_frame_error_threshold;
unsigned int           mac_frame_error_ratio;
unsigned int           mac_rmt_state;
unsigned int           mac_da_flag;
unsigned int           mac_una_da_flag;
unsigned int           mac_frame_error_flag;
unsigned int           mac_ma_unitdata_available;
unsigned int           mac_hardware_present;
unsigned int           mac_ma_unitdata_enable;
unsigned int           path_tvx_lower_bound;
unsigned int           path_t_max_lower_bound;
unsigned int           path_max_t_req;
unsigned int           path_configuration[8];
unsigned int           port_my_type[2];
unsigned int           port_neighbor_type[2];
unsigned int           port_connection_policies[2];
unsigned int           port_mac_indicated[2];
unsigned int           port_current_path[2];
unsigned char          port_requested_paths[3*2];
unsigned int           port_mac_placement[2];
unsigned int           port_available_paths[2];
unsigned int           port_pmd_class[2];
unsigned int           port_connection_capabilities[2];
unsigned int           port_bs_flag[2];
unsigned int           port_lct_fail_cts[2];
unsigned int           port_ler_estimate[2];
unsigned int           port_lem_reject_cts[2];
unsigned int           port_lem_cts[2];
unsigned int           port_ler_cutoff[2];
unsigned int           port_ler_alarm[2];
unsigned int           port_connect_state[2];
unsigned int           port_pcm_state[2];
unsigned int           port_pc_withhold[2];
unsigned int           port_ler_flag[2];
unsigned int           port_hardware_present[2];
       };
```

This structure, defined in `ccMib.h`, is used to return the results of `ccfddiGetStats`. A detailed explanation of the meanings of these fields may be found in the ANSI document *FDDI Station Management (SMT)*. Note that these definitions differ slightly from those defined in RFC-1512 (*FDDI Management Information Base*).

## 4.3 Application Example

Included in the driver distribution is sample source code (`ShowStat.c`) which shows how to use the ccfddiGetStats function to retrieve the FDDI MIB information.

The program simply prints out the contents of the MIB. Included in the source code are two functions,`ccfddiTime2Sec` and `ccfddiTimerTwosComp2Sec`, which demonstrate how ANSI time values are interpreted.

## 5. Getting Started

1. Boot VxWorks and load the ccFddiEnd*xxx*.a[1] lib with '`ld < ccFddiEndxxx.a`'.

2. Type '`muxDevStart ( muxDevLoad ( unitno, fddiLoad,"",0,0 ))`' to attach and start the driver. Verify by using `muxShow`. The *unitno* should usually be 0(Zero)[2]. If two FDDI cards are installed on the same host, unitno=0 for the first card and unitno=1 for the second card.

3. Attach the Driver to the TCP/IP protocol stack. Type '`ipAttach unitno, "ccfddi"' Use muxShow and ifShow` to confirm that the ccfddi*unitno* exists and was attached to TCP/IP.

4. Set the IP address using '`ifAddrSet`'. Refer to the VxWorks Programmers Guide paragraph '5.2.5 TCP/IP Internet Protocols and Addresses' and paragraph '5.3 Configuring the Network' for setting up IP host names and IP routing.

5. Confirm the FDDI connection using ping. An example is shown below.

   Note : This example assumes there is already a Local Area Network (NIC) on the FDDI Local Area Network (LAN) which has been set up with an IP address of 10.0.0.1.

```
-> muxDevStart ( muxDevLoad ( 0,fddiLoad, "", 0, 0 ))
value = 0 = 0x0

-> ipAttach 0 , "ccfddi"
value = 0 = 0x0

-> ifShow
ppp (unit number 0):
      Flags: (0x71) UP POINT-TO-POINT ARP RUNNING
      Internet address: 172.16.0.2
      Destination Internet address: 172.16.0.1
      Netmask 0xffff0000 Subnetmask 0xffff0000
      Metric is 0
      Maximum Transfer Unit size is 1 500
      5 packets received; 5 packets sent
      0 input errors; 0 output errors
      0 collisions
lo (unit number 0):
      Flags: (0x69) UP LOOPBACK ARP RUNNING
      Internet address: 127.0.0.1
      Netmask 0xff000000 Subnetmask 0xff000000
      Metric is 0
      Maximum Transfer Unit size is 4 096
      0 packets received; 0 packets sent
      0 input errors; 0 output errors
      0 collisions
ccfddi (unit number 0):
      Flags: (0x63) UP BROADCAST ARP RUNNING
      Netmask 0xffffffff Subnetmask 0xffffffff
      Ethernet address is 00:00:5a:45:f0:46
      Metric is 0
      Maximum Transfer Unit size is 4 491
      18 packets received; 0 packets sent
      0 input errors; 0 output errors
      0 collisions
value = 18 = 0x12
```

[1] *xxx*=178 for SVME178 PPC, 179 for SVME179, X86 for Intel X86 or Rad for Radstone PPC1a

[2] Known Bug In SENS1 under Tornado 1 :
You can not have more than one driver with the same *unitno*. So if you have an ethernet driver installed as ether0, you can not have a ccfddi0, but must make the *unitno* 1. This will change it to ccfddi1. This bug has been fixed in Tornado 2.

```
-> ifAddrSet "ccfddi0","10.0.0.4"
value = 0 = 0x0

-> ping "10.0.0.1"
PING 10.0.0.1: 56 data bytes
64 bytes from 10.0.0.1: icmp_seq=1. time=0. ms
64 bytes from 10.0.0.1: icmp_seq=2. time=0. ms
64 bytes from 10.0.0.1: icmp_seq=3. time=0. ms
64 bytes from 10.0.0.1: icmp_seq=4. time=0. ms
64 bytes from 10.0.0.1: icmp_seq=5. time=0. ms
```

## 6. Contact Details

### 6.1 Contact Person

Direct all correspondence and / or support queries to the Project Manager at C²I² Systems.

### 6.2 Physical Address

C²I² Systems
Unit 3, Rosmead Place, Rosmead Centre
67 Rosmead Avenue
Kenilworth
Cape Town
7708
South Africa

### 6.3 Postal Address

C²I² Systems
P.O. Box 171
Rondebosch
7701
South Africa

### 6.4 Voice and Electronic Contacts

| | | |
|---|---|---|
| Tel | : | (+27) (0)21 683 5490 |
| Fax | : | (+27) (0)21 683 5435 |
| Email | : | info@ccii.co.za |
| Email | : | support@ccii.co.za |
| URL | : | http://www.ccii.co.za/ |

### 6.5 Product Support

Support on C²I² Systems products is available telephonically between Monday and Friday from 09:00 to 17:00 CAT. Central African Time (CAT = GMT + 2).